BAŞLANGIÇ

▼ C# nedir?

- Microsoft tarafından .Net çatısı altında geliştirilen ve geliştirilmeye devam eden modern programlama dilidir.
- Açık kaynak ve ücretsizdir.
- C benzeridir ve OOP'yi destekler. **oop**→ programlamada bir yaklaşımı ifade eder ve problemleri nesneler olarak adlandırılan birimlere bölerek çözümlemeyi amaçlar
- Derlenen bir programlama dilidir. (Bunu sonra öğrenicez.)
- Web uygulamaları, mobil, web servis, dll, oyun....

▼ C# daki # işareti nerden gelir?

- C#, C++'ın bir üst versiyonudur. ++ operatörü, bir attırır o yüzde C++ ++ şekilinde gösterilmiştir. Burdan dies işareti ortaya çıkmıştır.

▼ .NET Framework ve .NET Core Nedir? Farkları Nelerdir?

- .Net: Microsoft'un developerlar için geliştirdiği teknolojileri sunduğu bir çatıdır.Bu çatı altında:
  - Desktop,Cloud, Game, machine learning..
- .Net Framework: Core'dan önce çıkmıştır. ve yalnızca Windows'da ayağa kaldırılabilir.
- .Net Core: Win bağımlılığını ortadan kaldırır. MacOs, Linux... hepsinde ayağa kaldırılır. Open source'dur.
  - C#, .Net Core'un bir alt koludur.
  - Modüler yapılandırma ile gelmiştir.

▼ Compiler nedir?

Eğer kullandığımız dil derlenen bir dilse, bu programlama dilinin önce derlenmesi, sonra çalışması, en son ise bir çıktı vermesi gerekmektedir.

c# ı ben anlayabiliyorum, derlendikten sonra makine anlayabiliyor. O yüzden bizim yazdığımız kodların derlenmesi gerekiyor.

Biz C# da Assembly yazmadan bilgisayarla haberleşmiş oluyoruz.

kodun kullanılabilir hale gelmesini sağlamaktayız derleme işlemi ile

- Bizim yazdığımız c# kodlarını makinenin öğrenmesi gerekir. Yazdıklarımızın makinenin anlayabilmesi için yazdığımız kodu derlemeliyiz
- Bilgisayarın anlayabileceği dile çevriliyor, bu assembly dilidir.
- .exe uzantılı dosya:Kullanıcının çalıştırabileceği yazılım uzantısıdır. Bu dosyayı derleme sonucunda alırız.
- Derlemenin ilk çıktısı .exe, diğer çıktısı dll'dir.

▼ Bir derleme nasıl yapılır? Mantığını anlayalım.

- Dosya uzantısı bizim için önemli değildir. Önemli olan dosya içerisindeki kodların satır satır okunabilmesidir.
- Developer command prompt(console uygulmasıdır.) ile yazdıklarımız(.net türevleri dosyalarımız) derlenebilir.
- cd komutu ile dosyanın bulunduğu dizine gidilir.
- cd dosya dizini \ csc dosya adı → enter'a bastığında o dosya dizinine .exe uzantısı gelir.

▼ Visual studio ortam tanıtımı

- Mimariler, codelar her şey. Tek seferde derleme yapılabilir. Derlemeler vs'e aittir.
- Community kişisel projelerini vs geliştirebilirsin. Professional ve enterprice takım çalışmalarında
- VS'de önemliler: Debug, tools, view→ solution explorer(Üzerinde çalıştığımız tüm dosyalar),view→ error list (Derleme sonucundaki hatalarımız)

▼ 6) vs Proje ve solution kavramları

- Proje: içerisinde amaca dair çözümler ve kodsal çalışmaların yapıldığı(yani operasyonların yürütüldüğü) bir bütündür. (Bankamatik)

- Solution: İçerisinde bir veya birden fazla proje barındırabilrn bir evransel kümedir. (Banka)
- Build: derleme yapar
- Rebuild: Önceden derlenip çıktı alınan dosyaları siler yeniden derler
- Clean: Derlenen dosyaları siler.
- Build denilince output kısmı açlır.
- Derlenen dosya nereye gider?
  - Projeye sağ tık
  - Open folder in File Explorer → Projenin olduğu dosya dizinini açar.
  - bin → Debug →netcoreapp3.1 gibi bir dosya bulunur. Bu dosya içerisinde .exe ve .dll çıktıları bulunur.
- Set a start up project. denildiğinde solution içerisinde ilk çalıştıırılacak olan proje seçilmiş olur

▼ 7) vsCode ortamı

- Kendi terminali bulunur. Bu terminalde ilerde CLI asistanını kullanabiliyor olacağız.
- Backend programlama yapılır.
- Build'i CLI dediğimiz kod üzerinde yaparız.

▼ 8) Dotnet CLI nedir? Temel komutları nelerdir?

- .Net komut satırı arayüzüdür.(Konsoldan talimat alır.)Nedir bu? bir asistandır
- Uygulamayı geliştirmeyi, oluşturmayı, çalıştırmayı ve yayınlamayı sağlar.
- .Net SDK'sı ile gelir.
- ! Talimatı verdiğin dizin çok önemli buna dikkat et. Çünkü ilgili projeyi o dizine oluşturur.
- cmd → temel komut satırlarını çalıştırır
- powershell →.NET Framework üzerine kuruludur ve çok daha geniş bir işlevsellik sunar.

- dotnet CLI temel KOMUTLAR:
  - `dotnet` → Dotnet CLI'ı çağırır. Bundan sonra talimat veririz.
  - `dotnet - -help` yazdığımızda bilmiyorsak ne yapacağımızı bize option verir.
  - `dotnet new` dediğimizde proje oluşturur. Enter dersek burda bize neler yapabileceğimizi gösterir.
    - dotnet new [projectType] - -name [projectName]
  - `dotnet new [projectType] - -name [projectName] - - force →` force parametresi ile verdiğimiz projectname daha önce verdiğimiz birproje adıyla çakışıyosa force bunu zorunlu hale getir.
  - `dotnet restore` komutu proje süresince referans edilen veya referansı kaldırılan paketlerin restoransını sağlar.
    - Referns nedir? Yazılımın gelişim süresince daha önceden yazılmış kütüphanelerin desteğini almaktır.
  - `dotnet build` dediğimizde projeyi build eder.
    - Derlemeden önce projeyi restore eder.
  - `dotnet publish` komutu projeyi yayınlayabilir hale getirmemizi sağlar. Projenin kaynak kodlarını saklamamızı sağlar.
    - Proje önce restore→build→publish Önce derler sonra bazı çıktılarını verir. Çıktı olarak:
      - .dll
      - .deps.json (projenin tüm bağımlılıklarını içerir.) Dependence
      - .runtimeconfig.json (runtime konfigürasyonları)
    - bin\debug\net5.0\publish dizininde çıktı verir.
  - `dotnet run` projeyi derledikten sonra ayağa kaldırır
    - dotnet run - - no- build parametresi projeyi derlemeden çalıştırır

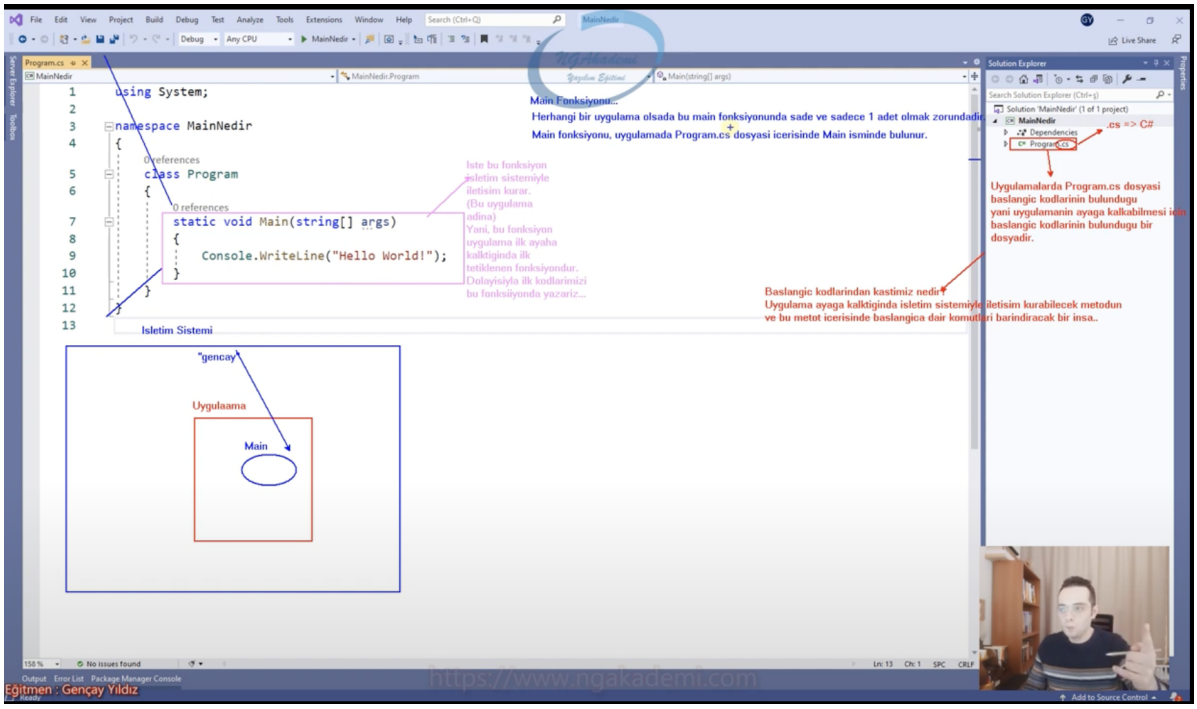
▼ 9) Proje modifikasyon komutları

Package: (Kütüphane) Nugget dediğimiz bir havuzdan gelir. İnternetten alınanlar vs.

Referance: önceden yazılmış dll'lerdir. Dosyanı bir başka yazılım kullanıcaksa .dll kullanılır. Ama eğer bir insn kullanacaksa, .exe dosyasını kullanabilir.

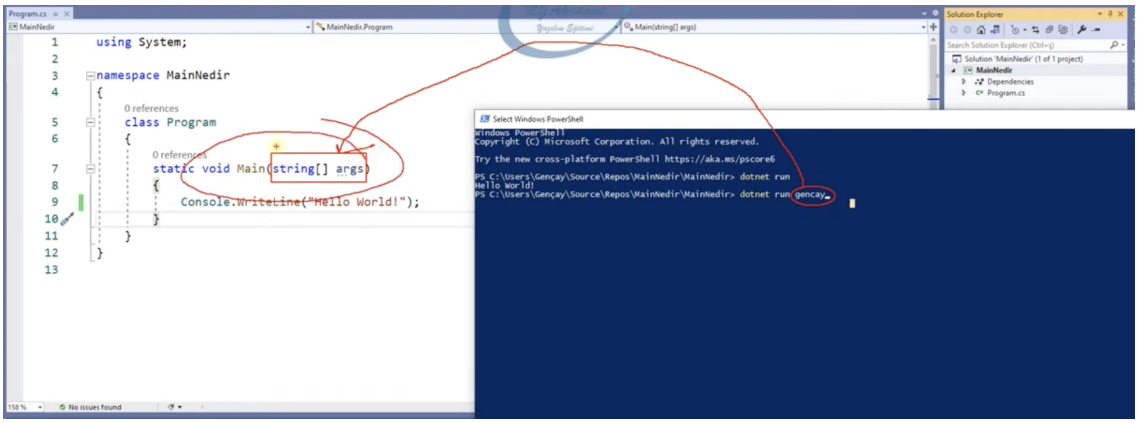
- add Package: Uygulamaya Nuget'ten paket/kütüphane yüklememizi sağlar.
- remove Package: add package ile eklediğmizi sileriz.
- add reference: Uygulamaya fiziksel bir dll dosyası referans etmemizi sağlar.
    - dotnet add [kaynak proje].csproj reference [hedef proje].csproj
- remove reference: add reference ile ekledikleirmizi siler.
- list reference: reference edilen tüm paketleri listeler.

▼ 10) Main fonksiyonu nedir?

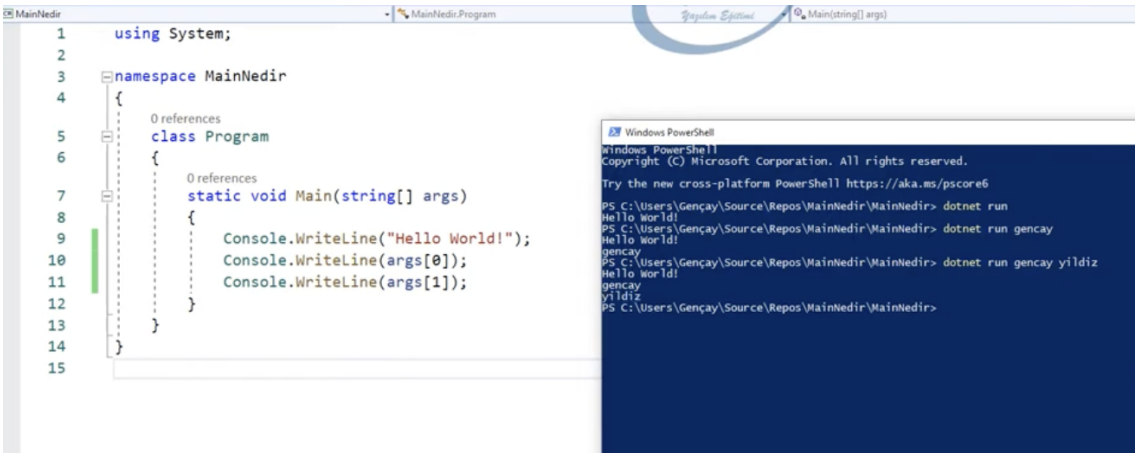


▼ 11) Main fonksiyonu işletim sisteminden nasıl bilgi alır?

- İlk adım olarak powerShell'de dotnet run xDeğeri ni çalıştırırsak:



- Sonrasında yeniden çalıştırdığımızda işletim sisteminden alabiliriz.



▼ Top level statement özelliği nedir?

- c# 9.0 ile ufak efek işlemlerde developerları yormamak için using System; komutundan sonra varsayılan/ basmakalıp (boilerplate) olarak gelen kodları yazmadan(yok sayarak) işlemlerini çalıştırabilir.
- Main fonksiyonunu(boilerplate) kullanmak developerın inisiyatifine kalmıştır bu özellik ile.
- Genellikle microservices yapılanmasında kodun gelişimi açısından hız kazandırıcı bir niteliğe sahiptir.
- Bu özelliğin kuralları:
  - Using blokları ile namespace arasında kodlar yazılabilir.

    - Bu işlem sadece Program.cs dosyasında geçerlidir. Farklı bir dsoyada gerçekleştirilmez.

▼ Yorum satırları ve region

- Yorum satırı: Kod konseptini ve sematik akışı bozmayacak şekilde istediğin her yerde kullanılır.
    - // ve /* lsjflnsdfl*/ şeklinde
- Region: kod dosyasını kategorik hale getiren ön işlemci komutudur.
    - #region ve #endregion arasına yazılır.

▼ To-Do özelliği nedir?

- yorum satırı yazarken //todo yorumyorumyorum şeklinde yazarsak view→ TaskList pencersi altına bu yorum gelecektir.

▼ Debbuging nedir?

- Programın hatalarını yok etmeye yönelik, yazılım kodu gözden geçirme , düzeltme aktiviteleridir. Hata ayıklamaktır.
- Debug modda'da ayağa kaldırılan bir uygulamada bir kodu nasıl debug ederiz?
    - Takip etmek istediğimiz kod blogunun başına breakpoint koyarız F9 ile.
    - O bloğa geldiğinde,
        - F10'a basılırsa adım adım ilerler.
        - F5'e bastığımızda uygulama devam eder, çalışır.
    - Add watch penceresi ile debug yaparken değişen kısımları inceeyebiliriz.
    - ctrl+F5 yapıldığında debug işlemleri yapılmadan uygulama ayağa kalkar.
    -